Deep F: Growing Deeper Roots

by: Kyran Flynn, Hannah Gross, and Shalin Patel
CSCI 1470: Deep Learning, Department of Computer Science

Introduction

Random Forests are an interesting candidate for deep learning as they are known for their stability in training, lack of hyperparameter tuning, good performance with little data, and high interpretability. The only previous attempt to use deep learning with Random Forests we found used feed forward networks for feature selection. We, however, applied a small feed forward network to learn a nonlinear split of the data at each node up the leaves on a subset of features. With this addition we looked to get the best of both worlds to provide both a high-performing and a simple, interpretable deep learning classification model.

Methodology

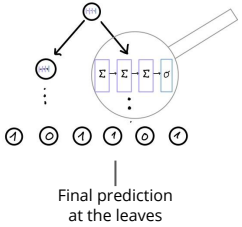
The core of the model is a decision tree where the dataset is queried according to a function of its features and splits the data to either the left or right child. Each leaf is associated with a final label, the classification of the examples that reach it. In a simple Random Forest, this split is learned by considering a randomly selected subset of the feature space and choosing the single feature and split point such that gini impurity is minimized.

In our model we take this randomly selected subset and pass it through an MLP with two output classes that determine a probability distribution on whether a given sample goes left or right (the figure below). Then, we use the following loss function to optimize our entire tree (and forest).

$$L(x, y \mid \theta) = \sum_{i=1}^{T} \sum_{\mathcal{N} \in \mathcal{T}_i} L_{\mathcal{N}}(x, y \mid \theta_{\mathcal{N}})$$

$$L(x, y, \mid \theta_{\mathcal{N}}) = h(s(x)_l \cdot y, \text{mode}(y_l)) + h(s(x)_r \cdot y, \text{mode}(y_r))$$

Here, N represents a node in the tree while $s(x)_k$ is the probability of the example going in the k^{th} direction. Similarly, y_k represents the labels of examples going to the k^{th} direction. We also have h as the cross entropy loss. Note that all y are in onehot form. We used the same set of hyperparameters for each dataset. Namely, we had 100 trees, each tree split on ¼ of the features, and the MLP splitter had a hidden size of 10 units.



For the interpretation task, we stuck close to the original Random Forest formulation where the loss in gini impurity across all nodes in the forest is aggregated per feature to give importance. In our case we used Shapley scores to determine a per-node relative importance in the split network for the feature subset at that node. We then weighted this with the inverse of the loss at the node and aggregated by feature for the entire forest to get importances.

Classification Datasets

The following datasets were chosen prior to any trials:

- Synthetic: Sine curve
- UCI ML Repositories: Iris, Wine, Breast Cancer

Results

For the sine curve example dataset (classification into points above and below the sine curve), our deep forest was able to capture the nonlinear partition of the dataset in a way that the random forest clearly failed to, as shown in figure 1.

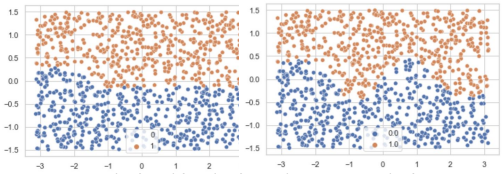


Figure 1: random forest (left) vs. deep forest (right) sine curve point classification.

Additionally, by multi-threading the training process of the deep forest, we were able to cut runtime from around 45 minutes (unthreaded) to 6 minutes (threaded) on an eight-core Macbook Pro for our largest dataset (ICI Breast Cancer).

On all the datasets, we did at least as well and mostly significantly better as compared to the benchmark. The summary of our results relative to our benchmark models are shown below, in figure 2.

Dataset	Random Forest	MLP	Deep Forest
Sine Curve	0.934	0.967	**0.984**
Iris	0.96	**0.987**	**0.987**
Wine	0.983	0.983	**0.994**
Breast Cancer	0.965	0.627	**0.979**

Figure 2: results by datasets and model type. Highlighted is the best classification accuracy.

We also saw similar weighting of feature importances in our deep forest as the random forest benchmarks, indicating that our model is similarly able to demonstrate meaningful weighting of features, as seen in figure 3 below for comparison.

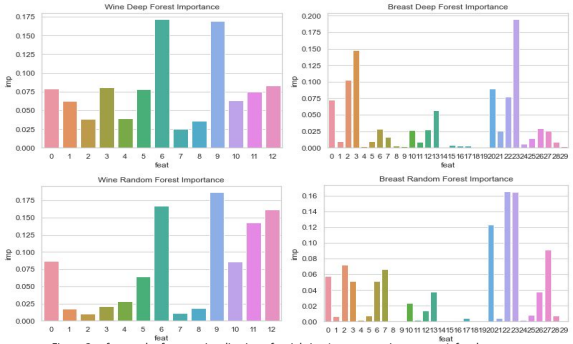


Figure 3: a feature-by-feature visualization of weighting (percentage importance), for deep forests (top) and random forests (bottom), on our wine and breast cancer datasets.

Discussion

We are pleased with our results. On classic benchmarking datasets our model outperformed the standard random forest, and on synthetic datasets designed to play to a deep forests' potential strengths, it outperformed random forests significantly. Not only that, but in almost all the datasets it outperformed a baseline neural net.

Another interesting feature of the model is that it has no tendency to overfit the data. This is due to the fact that the loss function never directly takes the loss based on the labels. Instead, the loss is focused on splitting the dataset as effectively as possible and creating purer and purer nodes.

With regards to environmental impact, this model needs less calibration because it doesn't overfit and also is to a large extent insensitive to hyperparameter changes - leading to less runs.

Furthermore, this model provides very good interpretation results. As can be seen in figure 3, the Deep Forest picks up on the most important features in the datasets such as the Wine and Breast datasets in comparison to the Random Forest importances. Since Random Forests have been shown to have good interpretation, the interpretations that are generated by Deep Forest are both relevant and useful.

Overall, Deep Forest seems like a promising model that is more performant than its MLP and Random Forest counterpart and is able to deliver interpretation results